CS 134: Networks	Problem set 8
Prof. Yaron Singer	Due: **Wednesday 4/5/2017**

1. Alternative Definition for Submodular Functions. Recall that in class we said that a set function $f : 2^V \to \mathbb{R}$ is *submodular* if for every $S, T \subseteq V$ for which $S \subseteq T$ and $a \notin T$ we have that:

$$f(S \cup \{a\}) - f(S) \geq f(T \cup \{a\}) - f(T)$$

a. Show that f is a submodular function **if and only if** for every $A, B \subseteq V$ we have that:

$$f(A \cup B) \leq f(A) + f(B) - f(A \cap B)$$

b. Show that every nonnegative submodular function is subadditive, i.e. if f is submodular then $\forall A, B \subseteq V$:

$$f(A \cup B) \leq f(A) + f(B)$$

Solution:

Answer: First, I will show this is true in the forwards direction. Let us say the following:

$$S = A \cap B$$

$$a = A \setminus B$$

$$T = B$$

A function is submodular if the following statement holds:

$$f(S \cup a) - f(S) \geq f(T \cup a) - f(T)$$

This can also be rewritten as follows:

$$f(S + a) - f(S) \geq f(T + a) - f(T)$$

Plugging in for my definitions of S, a, and T, the following is true:

$$\begin{aligned} f((A \cap B) + (A \setminus B)) - f(A \cap B) &\geq f(B + (A \setminus B)) - f(B) \\ = f(A) - f(A \cap B) &\geq f(A \cup B) - f(B) \\ = f(A \cup B) &\leq f(A) + f(B) - f(A \cap B) \end{aligned}$$

Now I will show the backwards direction. Let us saying the following:

$$A = S \cup a$$

$$B = T$$

We start with the following:

$$f(A \cup B) \leq f(A) + f(B) - f(A \cap B)$$

Plugging in for my definition of A and B, the following is true:

$$f((S \cup a) \cup T) \leq f(S \cup a) + f(T) - f((S \cup a) \cap T)$$

Because $S \subseteq T$, the following is true:

$$f(a \cup T) \leq f(S \cup a) + f(T) - f(S)$$

$$= f(S \cup a) - f(S) \geq f(T \cup a) - f(T)$$

b. (8 points) Show that every submodular function is subadditive, i.e. if if f is submodular then $\forall A, B \subseteq V$:

$$f(A \cup B) \leq f(A) + f(B)$$

We have just shown in 1a that the following is true for submodular functions:

$$f(A \cup B) \leq f(A) + f(B) - f(A \cap B)$$

Because we know that $f(A \cap B)$ is greater than or equal to 0, removing the $f(A \cap B)$ part of the equal still holds true. For example, if $f(A \cap B)$ is some very large number, we know that $f(A \cup B)$ would still be less than or equal to $f(A) + f(B) - f(A \cap B)$. Without the $f(A \cup B)$ term, this inequality must still hold true.

2. The Transition Matrix (10 points) Given a graph $G = (V, E)$, for any $u \in V$, let $\mathcal{N}(u)$ denote the set of neighbors of u, including u itself, and $d(u) = |\mathcal{N}(u)|$. Recall that the *transition matrix* of G is the $n \times n$ matrix M defined as:

$$M_{u,v} = \begin{cases} 1/d(u) & v \in \mathcal{N}(u); \\ 0 & \text{otherwise} \end{cases}$$

Consider the following network

$$G = (\{a, b, c, d, e, \}, \{ab, bc, cd, de, ac, ad, ae, aa, bb, cc, dd, ee\}).$$

a. **(6 points)** Construct the transition matrix for G.

b. **(7 points)** What is the probability of a random walk going from node a to node e in two steps?

c. **(7 points)** In the voter model, what is the probability that the opinion of node a reaches e in two steps?

Solution by Matthew Huang:

2. (a)

$$M = \begin{bmatrix} 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/4 & 1/4 & 1/4 & 1/4 & 0 \\ 1/4 & 0 & 1/4 & 1/4 & 1/4 \\ 1/3 & 0 & 0 & 1/3 & 1/3 \end{bmatrix}$$

(b) The probability of going from a to e in two steps is given by:

$$\mathbf{1}_a^\top M^2 \mathbf{1}_e = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/4 & 1/4 & 1/4 & 1/4 & 0 \\ 1/4 & 0 & 1/4 & 1/4 & 1/4 \\ 1/3 & 0 & 0 & 1/3 & 1/3 \end{bmatrix}^2 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \approx 0.157$$

(c) The probability that e adopts the opinion of a after two steps is:

$$\mathbf{1}_e^\top M^2 \mathbf{1}_a = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \\ 1/3 & 1/3 & 1/3 & 0 & 0 \\ 1/4 & 1/4 & 1/4 & 1/4 & 0 \\ 1/4 & 0 & 1/4 & 1/4 & 1/4 \\ 1/3 & 0 & 0 & 1/3 & 1/3 \end{bmatrix}^2 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \approx 0.261$$

3. Random Walks (10 points) Prove that for a graph $G = (V, E)$ with transition matrix M, for any $u, v \in V$ the probability of a random walk starting at u and ending at v after t steps is $\mathbf{1}_u M^t \mathbf{1}_v^\top$ where $\mathbf{1}_x$ denotes the row vector that takes the value 1 in the index that corresponds to the node $x \in V$ and 0 in all other indices.

Solution by Xiner Zhou: The proof is by induction on t.

For $t = 0$, the probability of a random walk starting at u and ending at v after 0 step is 1 if $u = v$, and is 0 otherwise:

$$p^0_{u,v} = \begin{cases} 1, & \text{if } u = v \\ 0, & \text{otherwise} \end{cases}$$

Since $\nVdash_{\approxeq} M^0 \nVdash^{\mathbb{T}}_{\succapprox} = \nVdash_{\approxeq} \nVdash^{\mathbb{T}}_{\succapprox} = \begin{cases} 1, & \text{if } u = v \\ 0, & \text{otherwise} \end{cases}$ Therefore, $p^0_{u,v} = \nVdash_{\approxeq} M^0 \nVdash^{\mathbb{T}}_{\succapprox}$.

For $t = 1$, the probability of a random walk starting at u and ending at v after 1 step is $\frac{1}{d(u)}$ if $v \in d(u)$, and is 0 otherwise:

$$p^1_{u,v} = \begin{cases} \frac{1}{d(u)}, & \text{if } v \in d(u) \\ 0, & \text{otherwise} \end{cases}$$

Since $\nVdash_{\approxeq} M^1 \nVdash^{\mathbb{T}}_{\succapprox} = \begin{cases} \frac{1}{d(u)}, & \text{if } v \in d(u) \\ 0, & \text{otherwise} \end{cases}$ Therefore, $p^1_{u,v} = \nVdash_{\approxeq} M^1 \nVdash^{\mathbb{T}}_{\succapprox}$.

For general $t > 1$, assume $p^{t-1}_{u,v} = \nVdash_{\approxeq} M^{t-1} \nVdash^{\mathbb{T}}_{\succapprox}$. The process that a random walk starting from u and ending at v after t steps, can be break down into two stages: on the first stage, a random walk starts from u and ends at one of its neighbors w after 1 step, with equal probability $\frac{1}{d(u)} = \frac{1}{|N(u)|}$; on the second stage, a random walk starts from w and ends at v after $t - 1$ step. Therefore:

$$\begin{aligned}
p^t_{u,v} &= P[a\ random\ walk\ from\ u\ to\ v\ after\ t\ steps] \\
&= \sum_{w \in N(u)} \frac{1}{|N(u)|} P[a\ random\ walk\ from\ u\ to\ w\ after\ 1\ steps] \\
&\quad \times P[a\ random\ walk\ from\ w\ to\ v\ after\ t-1\ steps] \\
&= \sum_{w \in N(u)} \frac{1}{|N(u)|} p^{t-1}_{u,w} p^1_{w,v} \\
&= \sum_{w \in N(u)} \frac{1}{|N(u)|} \nVdash_{\approxeq} M^{t-1} \nVdash^{\mathbb{T}}_{\succapprox} \nVdash_{\approxeq} M \nVdash_{\succapprox} \\
&= \sum_{w \in N(u)} \frac{1}{|N(u)|} \nVdash_{\approxeq} M^{t-1} M \nVdash_{\succapprox} \\
&= \sum_{w \in N(u)} \frac{1}{|N(u)|} \nVdash_{\approxeq} M^t \nVdash_{\succapprox} \\
&= \nVdash_{\approxeq} M^t \nVdash_{\succapprox}
\end{aligned}$$

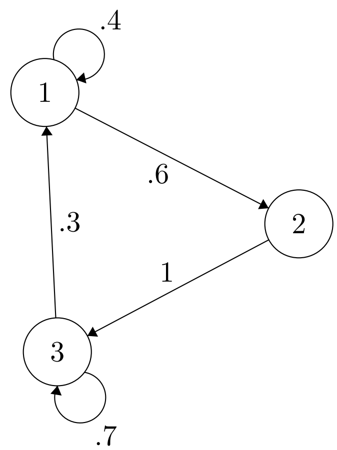
Thus, by induction, the probability of a random walk starting from u and ending at v after t steps is $p^t_{u,v} = \nVdash_{\approxeq} M^t \nVdash_{\succapprox}$.

4. DeGroot Model (30 points) Consider a group of n friends. At time $t = 0, 1, 2, \ldots$ each person i has an opinion, $p_i(t) \in [0, 1]$, of the new James Bond movie. Every period they talk to each other and exchange opinions. The *DeGroot model* posits that person i puts weight $m_{ij} \geq 0$ on person j's opinion (putting weight on one's own opinion is allowed), subject to $\sum_{j=1}^{n} m_{ij} = 1$. In other words, if in period t, the people's opinions are $(p_1(t), ..., p_n(t))$, then i's opinion in period $t + 1$ is

$$p_i(t+1) = \sum_{j=1}^{n} m_{ij} p_j(t) \tag{1}$$

a. **(3 points)** Show that Eq. (1) can be rewritten as $p(t + 1) = Mp(t)$, where $p(t + 1)$ and $p(t)$ are vectors of length n and M is a $n \times n$ matrix.

b. **(3 points)** Consider the network given in the picture below. An arrow from node i to node j means that person i puts weight on j's opinion. Write down M.



c. **(4 points)** It turned out that person 1 loved the movie, person 2 thought it was pretty good, and person 3 didn't get it: $p(0) = (1, 0.7, 0)$. Find $p(1)$ and $p(2)$.

d. **(6 points)** What are the limiting opinions? In other words, find $\lim_{t\to\infty} p(t)$. Does the group reach a consensus? Give an intuitive explanation of why consensus is reached.

e. **(7 points)** Give an example of a network with $n = 3$ with no isolated nodes, and a vector of initial opinions, such that a consensus is never reached – that is, so the sequence of vectors $(p(t))_{t=1}^{\infty}$ has no limit.

f. **(7 points)** Suppose now agents are arranged in a ring, numbered $1, 2, \ldots, n$ with person i putting $1/2$ of her updating weight on himself, $m_{ii} = 1/2$, and $1/2$ of her weight on his clockwise neighbor, $m_{i,i+1} = 1/2$. Indices are read modulo n.

What will opinions converge to now? I.e., write down an explicit formula for $p(\infty)$, which is defined to be $\lim_{t\to\infty} p(t)$. This formula should involve only entries of $p(0)$ and numbers. Justify your answer. This can be done without any calculation.

Solution by Jack Wang:

a) Let $p(t)$ be an vector of opinions where the ith entry correspond's to person i's opinion at time t. Let M be an $n \times n$ matrix where each entry m_{ij} corresponds to person j's influence on person i. Then,

$$p(t+1) = Mp(t)$$

and each individual entry $p_j(t+1) = \Sigma_{j=1}^{n} m_{ij} p_j(t)$

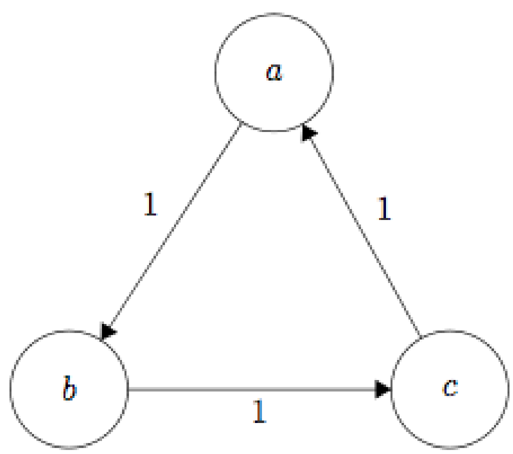
b)

$$M = \begin{bmatrix} .4 & .6 & 0 \\ 0 & 0 & 1 \\ .3 & 0 & .7 \end{bmatrix}$$

c) $p(1) = (.82, 0, .3)$, $p(2) = (.328, .3, .456)$

d) $\lim_{t\to\infty} p(t) = (.394, .394, .394)$. Yes, the group does reach a consensus. Intuitively this is because every person's opinion can be affected by every other person's opinion, and there is at least one person who gets their opinion from multiple sources.

e) With the following graph and initial opinions $(1, 0, 0)$ the positive opinion gets passed around infinitely.



f) Each opinion in $p(\infty)$ will converge to

$$\frac{\Sigma j = 1^n p_j(0)}{n}$$

In other words they will all take on the average opinion of the entire ring.

5. Learning Influence Locally (25 points) This problem is similar to problem 5 from problem set 6, but with a new diffusion model—the Voter model.

As in the previous problem, let's say we have a graph $G = (V, E)$ and some opinion data for each node $u \in V$. Assume each node has an opinion that changes over time based on the influence of its neighbors for a particular observed diffusion process. For a particular observed diffusion process i, let $opinion_i$ be a vector of timestamped opinions, where $opinion_i[u]$ is a vector such that $opinion_i[u][t]$ is equal to the opinion of node u at time t during observed cascade i, where t ranges from 0 to $\tau - 1$ so that $opinion_i[u]$ has length τ, where opinions are either 0 (node is inactive) or 1 (active).

In the Voter model, each directed edge (u, v), has some corresponding edge weight $w_{(u,v)} \in (0, 1]$. At each time step t, node u will look at each of its neighbors and update its opinion to reflect the majority "weighted" opinion—that is, if $\sum\limits_{v \in N(n), o_{t-1}(v)=1} w_{(u,v)} \geq .5$, then u will update its opinion to be 1 at time step t, and otherwise, will update its opinion to be 0 at time step t. In the summation, $o_{t-1}(v)$ refers to the opinion of node v at time step $t - 1$ and is either 0 or 1. How might we describe this situation using a linear program such that solving said linear program will give us an approximation for the edge weights?

If we want to fit the Voter model locally to the opinion data, we can learn each of the edge weights $w_{(u,v)}$ in the graph by modeling the problem as binary classification and solving using a linear program. Consider a node n. We're trying to learn each of the edge weights $w_{(u,v)}$ for $v \in N(u)$. If we have some opinion data, and we know that at some time step t, u has opinion 0, and at time step $t + 1$, u has opinion 1, we know that the sum of the edge weights to u's neighbors that had opinion 0 at $t - 1$ was $\geq .5$ at $t - 1$, but was $< .5$ at t.

The file *pset8_network1.txt* is a directed graph, where $a \quad b$ corresponds to an edge from a to b. The file *node_opinions.pk* is an array of opinions, where each entry *node_opinions*$[i]$ is a dictionary equivalent to the vectors $opinion_i$ described above, such that $opinion_i[n]$ gives the vector of timestamped opinions described.

a. **(1 point)** Load the file *pset8_opinions.pk*. If you're using python, you can use python's 'pickle' library and the 'load' function to easily turn the file into an array; if you're using another language, we've provided *node_opinions.json*, a JSON-encoded file that you can easily load in any language using the standard JSON encoding and the respective JSON library for your language (though you might have to do some additional string to integer conversion). What is τ and how many cascades are there?

b. **(1 point)** Now load the file *pset8_network1.txt*. How many nodes are there? What is the average (out-)degree?

c. **(10 points)** Using the definition of the Voter model described in lecture, and given a node u, formulate a linear program that, for node u, and given d diffusion processes with τ time steps each, uses all the information provided by the diffusion processes to approximate the edge weights for $w_{(u,v)}$ for $v \in N(u)$. Feel free to use indices to refer to time steps and cascades, but please be clear with your notation. Don't forget to also include constraints such as the fact that all the edge weights have to be nonnegative, or that $\sum\limits_{v \in N(u)} w_{(u,v)} = 1$.

d. **(10 points)** Using the linear program you have described above, solve the program for each node and approximate the edge weights of each $e \in E$. To solve the program, use one of the linear program solver packages available (scipy has a linear programming package). Note that linear program solver packages will find variables that satisfy the constraints provided and fit some maximization or minimization constraint. Typically, we want to minimize the sum of slack variables; however, in this case, we know that the Voter model can be fit to the data (since the data was generated by the Voter model), so we don't need to include slack variables. **For the sake of making everything standardized for the grader, just maximize the sum of all the variables you're using in the linear program**.

What's the estimated weight of edge $(2, 30)$? Of $(29, 5)$? Additionally, submit your estimates as a separate file formatted as a .csv, where entries in the first column correspond to the first node in the edge, entries in the second column correspond to the second node in the edge, and entries in the third column correspond to the edge weight.

e. **(3 points)** What node has the highest average edge weight (for outgoing edges)? What node has the lowest average edge weight for outgoing edges?

Solution by Lars Lorch:

a. $\tau = 35$ and there are 35 different diffusion processes.

b. There are 35 nodes in the network. The average out-degree is 3.3714.

c. Given a node u, d diffusion processes o^i with τ time steps each, the edge weights $w_{(u,v)}$ for $v \in N(u)$ have to be defined by the following conditions.

$$o_t(u) \in \{0, 1\}$$
$$\sum_{v \in N(u)} w_{(u,v)} = 1$$
$$w_{(u,v)} \geq 0$$

For all $t \in \tau$ and $o^i \in \mathbf{o}$ with $|\mathbf{o}| = d$,

$$o_t^i(u) = 0 \Rightarrow \sum_{v \in N(u)} w_{(u,v)} o_{t-1}^i(v) < 0.5$$
$$o_t^i(u) = 1 \Rightarrow \sum_{v \in N(u)} w_{(u,v)} o_{t-1}^i(v) \geq 0.5$$

d. The estimated edge weight of edge $(2, 16)$ is 0 and of edge $(29, 22)$ is 0.

e. The five students (nodes) that Raynor should invite first to the app are nodes 5, 3, 10, 6, 12 with respective average influences of 0.5, 0.464, 0.433, 0.375, 0.375. [Staff note: these influences can vary a lot depending on the specific implementation of the LP]